AKD Profinet Telegram (Cyclic Data)
Quick Reference Guide
Rev 6

12/13/2017

By Jimmy Coleman

This document is for a quick reference to some information about Profinet communications with the AKD drive. It contains notes about the telegram data (cyclic data), PLC setup, and Profinet Protocol. Please also reference the AKD Profinet Manual.

Baudrate:

Set the baud rate to “Automatic” for best results. The GSDML file sets the baud rate to 16ms. You can set the baud rate to 16ms with a “Fixed update time”, but if you do, the PLC will not reconnect to the drive after the drive is power cycled.

AKD command source settings:

The command source (DRV.CMDSOURCE) will be set automatically based on the Telegram number that is written to the drive.

- Telegrams 1, 350, and 351 use “Fieldbus” "Velocity Mode".
- Telegrams 7, 9, 352, and 400 use “Service” "Position Mode".

(The command source is based on where the actual motion command is generated. With the AKD in position mode and running motion tasks, the motion command is coming from the Command Generator in the drive. The PLC sends the values for the motion task, but the incremental/instantaneous command is from the drive itself. In velocity mode, the motion command is coming directly from the PLC.)

Selecting a Telegram:

The telegram is selected in the PLC, using the GSDML (xml) file. The telegram can be configured in the PLC's hardware configuration and sent to the drive during initialization. Or the telegram can be selected using PNU 922 and PNU 930 in the PLC program.

You can't select more than one telegram in the hardware configuration. So if you want to use two telegrams, one must be selected in the HW configuration and then switch telegrams using the PNU's in the program. But this can cause data length conflicts. I have successfully started with TG400 selected in the HW configuration and then switched between TG400 and TG1 in the program using the PNU's. It seems that the AKD doesn't care about the telegram data length. It will just use whatever data is in each word of the telegram that is set in the drive with PNU 922.

But the PLC function block will not allow specifying a telegram different from the one in the HW configuration. So just start with TG400 and then you will have all of the telegram words available for use with any telegram.

You can read the currently selected telegram number in FBUS.PARAM01. This parameter is read/write, but setting the value of FBUS.PARAM01 directly in Workbench does not select a telegram. It must be defined by the PLC.

Operation mode:

The standard telegrams offer control in Velocity mode and Position mode.

- Telegrams 1, 350, and 351 use Velocity mode.
- Telegrams 7, 9, and 352 use Position mode.

The operation mode is not configured automatically by selecting a telegram. The operation mode (DRV.OPMODE) must be configured in Workbench (or Telnet communication) or by using PNU 930 (or PNU 2135).
The operation mode MUST be configured while Control Word bit 10 is OFF.

Control Word Bit Sequence:
For the Control Word (STW1), the bits must be turned on in the proper sequence. It is a state machine. Each bit commands a transition from one state to another. After each bit is turned on, make sure your program keeps the bits turned ON for every update cycle. If the bits go low again, that will command a state transition.

The function block in the Sample Project controls the Control Word bits properly, so you don't have to do it in your ladder program. The following sequence is part of the theory of the Profinet State Machine for reference. The full state machine is defined in the AKD Profinet Manual.

Turn on bit 10 – enable the telegram data words. Establishes PLC control.
Turn on bits 1 and 2. The order doesn't matter.
Turn on bit 0 – "Switched-On"
Turn on bit 3 – "Operation Enabled" (drive enables, ready for motion commands)
Turn on bits 4 and 5, the order doesn't matter. These are for stop and pause.
Turn on bit 6 to start motion. But before you turn on bit 6, make sure you have the appropriate motion profile data entered into the telegram.

Command behavior of Telegram 1:
Control Word bit 6 enables the speed setpoint (starts motion).
Motion stops when bit 6 = 0.
Motion starts when bit 6 = 1.
Motion continues as long as bit 6 = 1.
If the speed setpoint is changed during motion, then the commanded speed will change. There is no need to toggle bit 6.

Command Values in Telegram 9:
STW1 Turn on bits: 10, 1 and 2, 0, 3 (enable), 4 and 5, 6 (start motion)
SATZANW1 - Set to Motion Task #, or set to 0x8000 to use MDI move profile values
STW2 - not used
MDI_TARPOS =DW#16#00010000 = 10000 hex = 65536 counts = 1 rev
MDI_VELOCITY =DW#16#02222222 = 100 rpm
MDI_ACC =W#16#000E = 14 dec = 10,681 rpm/s
MDI_DEC =W#16#000E = 14 dec = 10,681 rpm/s
MDI_MOD - 1 = absolute move; 0 = relative move; (TG9 profile only)

Velocity Mode Scaling: (for Telegrams 1, 350, 351)

- NSOLL_A (speed setpoint) = (Speed setpoint in rpm) * 2^15 / 12000

Speed setpoint in rpm = NSOLL_A * 12000 / 2^15

- NIST_A = Actual speed in rpm * 2^15 / 12000
  Actual speed in rpm = NIST_A * 12000 / 2^15

Example: for commanding 60 rpm
NSOLL_A = 60 * 2^15 / 12000 = 164 dec (0x00A4)

Example: for commanding -200 rpm
NSOLL_A = -200 * 2^15 / 12000 = -546 dec (0xFDDE)

Example: for reading actual speed in rpm (setpoint = 60rpm)
NIST_A = 0x00A7 (167 dec)
Actual speed in rpm = 167 * 12000 / 2^15 = 61 rpm

Example: for reading actual speed in rpm (setpoint = -200rpm)
NIST_A = 0xFDCF (-561 dec)
Actual speed in rpm = -561 * 12000 / 2^15 = -205 rpm

Position Mode Scaling: (for Telegrams 9, 352, 353, 400)

- Position resolution is set with PNU# 1002. Default is 16 for 2^16 counts per rev.

- MDI_Velocity = (Speed in rpm) * 2^32 / 12000
  Accel and Decel scaling can be changed with PNU 1008 (Default is 16 for 2^16 = 50,000,000 rpm/s).

- MDI_ACC = (accel in rpm/s) * 2^16 / 50,000,000
- MDI_DEC = (decel in rpm/s) * 2^16 / 50,000,000
  Accel and decel values will be in increments of about 763 rpm/s.

- NIST_A = Actual speed in rpm * 2^15 / 12000
  Actual speed in rpm = NIST_A * 12000 / 2^15

Example: for reading actual position
XIST_A = 0x1DE52EB (501558968 dec)
Actual Position = 501558968 counts, where 65536 counts is 1 rev.

Current Scaling: (for Telegrams 350 and 351)

- Actual current in Arms = ITIST_GLATT * DRV.IPEAK / 2^14

Example: for reading actual current in Arms
ITIST_GLATT = 0x036D (877 dec)
Actual current in Arms = 877 * 9 / 2^14 = 0.482 Arms

Example: for reading actual current in Arms (negative direction)
ITIST_GLATT = 0xF840 (-1984 dec)

Actual current in Arms = -1984 * 9 / 2^14 = -1.090 Arms

None of the standard telegrams use current command (signal # 102, IL_CMD_FIELDBUS, IL.CMD). However, Telegram 400 can be mapped for torque mode using the current command. Scaling would be: IL_CMD_FIELDBUS = IL.CMD * 2^14 / DRV.IPEAK, where IL.CMD is the desired current command.

AKD Parameters used in Workbench Terminal screen:

PN.STW1
PN.ZSW1
PN.POSSCALE
PN.TIMEOUTFTHRESH
PN.DIAG

Wireshark Display Filter:

Show only the PN-IO based traffic:
pn_io
Show the PN-IO based traffic without the cyclic PN-IO telegrams (to avoid a lot of "noise"):
pn_io && !pn_io.ioxs

ZSW2:

ZSW2 (status word 2) is a counter that continuously counts the number of telegram read cycles. This can be used by a PLC program to keep track of the number of telegram scan cycles.

Jog Moves:

There are two ways to do a jog move in the AKD through Profinet:

- STW1 bit 12 = 0 -- Use STW1 bits 8 or 9 to start a jog move, where the speed is defined by SM.V1 (for bit 8) and SM.V2 (for bit 9). The direction is based on the sign of the value of SM.V1 or SM.V2 (whichever is used). SM.V1 can be set in Workbench or with PNU 1004. SM.V2 can be set in Workbench or with PNU 1005.

- STW1 bit 12 = 1 -- Use STW1 bit 8 to start a jog move, where the speed is defined by MDI_VELOCITY of Telegrams 9, 352, 353, or 400. The direction is specified by STW1 bit 13. First set bit 12 to enable “real-time jog”. Then set bit 13 to specify the direction of motion. Then start the jog move with bit 8. The acceleration is specified by MDI_ACC and the deceleration is specified by MDI_DEC. If these are set to zero, then the axis will not move. The drive does not need to be homed prior to running a jog move.